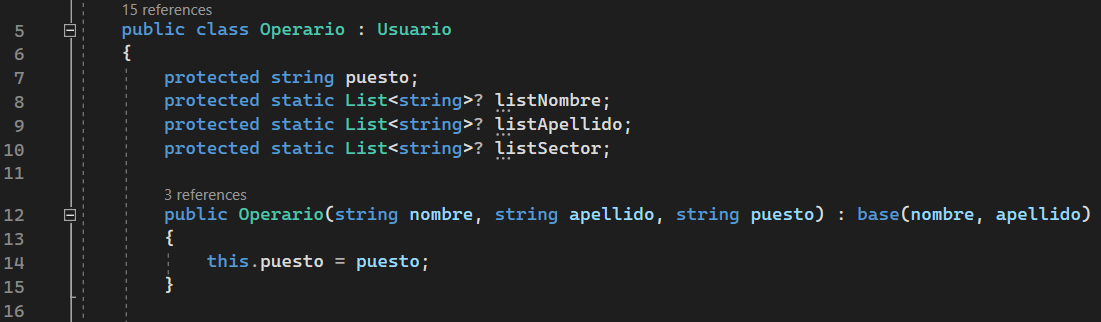
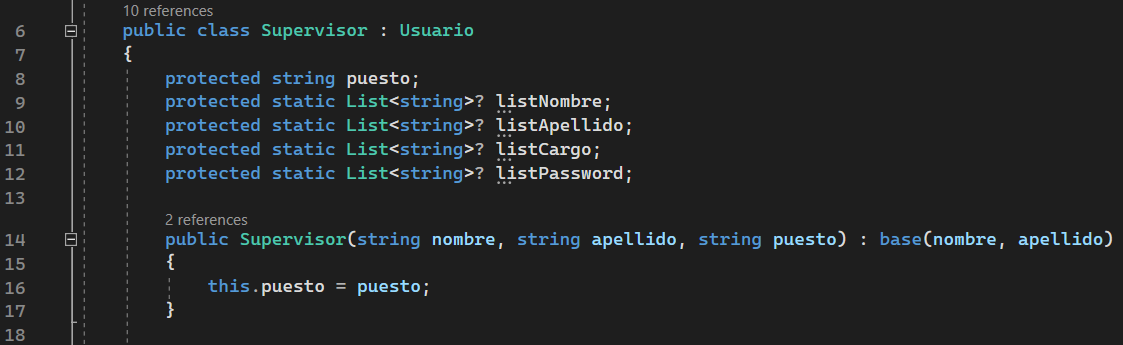
**Temas utilizados en el proyecto**

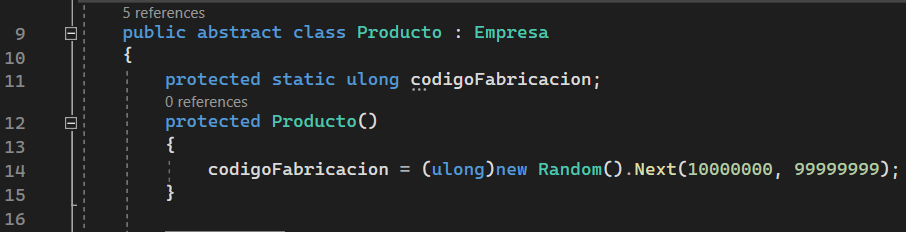
**+ Herencia:**

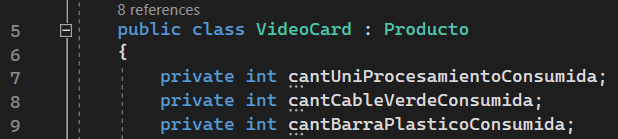
Aplico herencia en dos casos, uno de ellos es con las clases Operario y Supervisor que heredan de Usuario.





Luego tenemos la clase Producto que hereda de Empresa, y luego VideoCard, Motherboard, Cabinet y Ram que heredan de Producto.

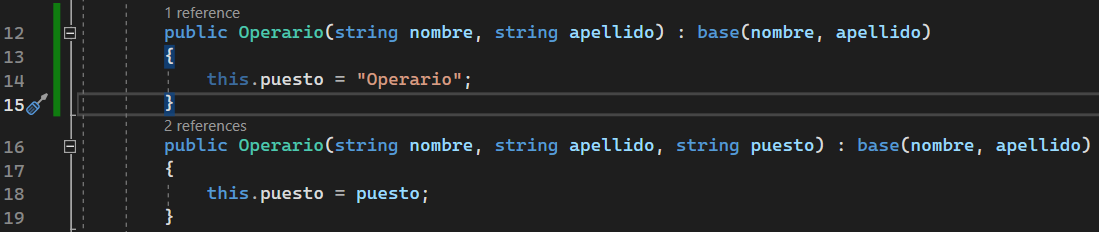


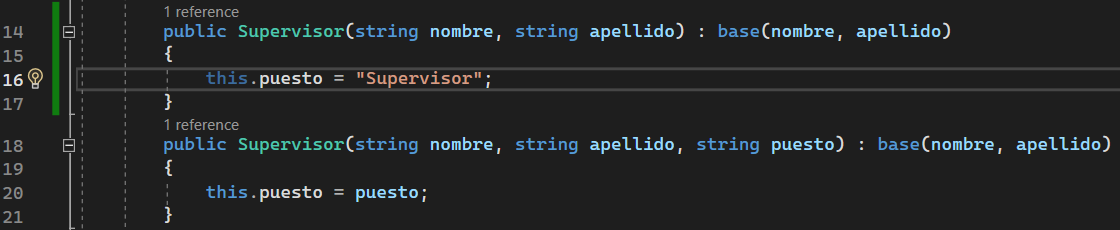


En ambos casos aplico herencia con el fin de obviar escribir tanto código, de esta manera es mas legible y práctico para trabajar con ellas, ya que mis clases derivadas contienen información de sus clases base y de esta forma me evito escribir tanto código innecesario.

**+ Sobrecarga:**

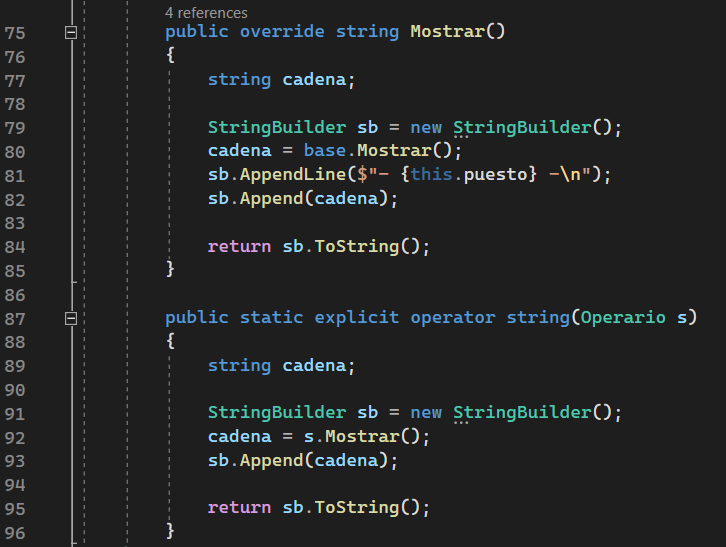
Aplico sobrecarga de constructores en las clases Operario y Supervisor.



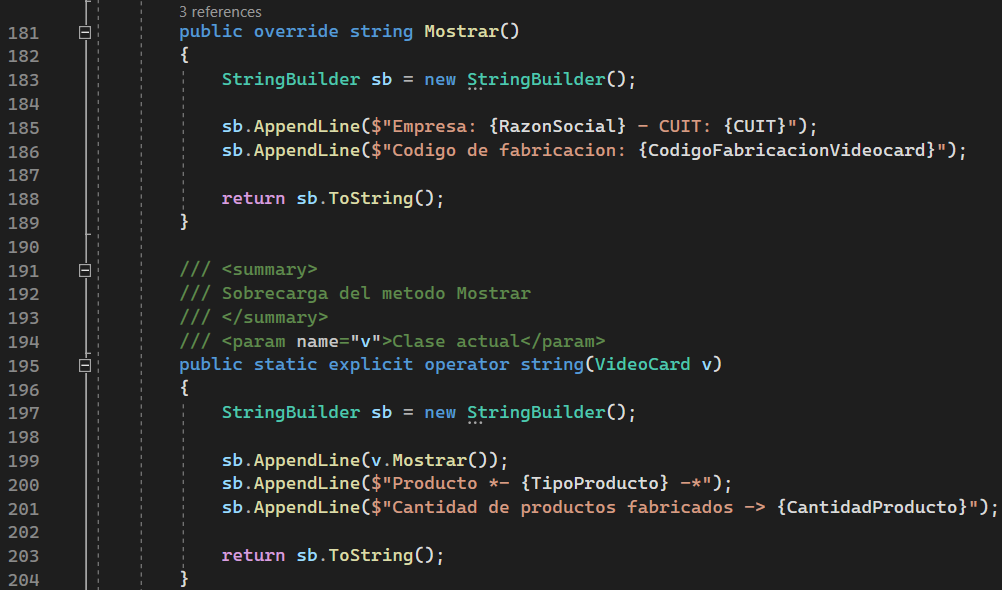


Al utilizar la sobrecarga en los constructores me permite tener más de una forma de instanciar una clase dándole más o menos parámetros por referencia. De esta forma puedo tener un objeto con más parámetros dados por el usuario o menos ya que tendría algunos datos escritos de "forma default".

Aplico sobrecarga de métodos en las clases Operario y Supervisor - y luego VideoCard, Motherboard, Cabinet y Ram.

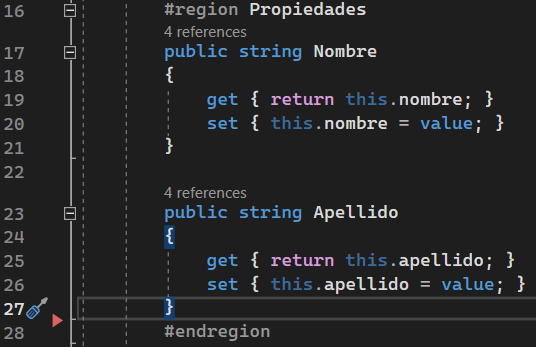


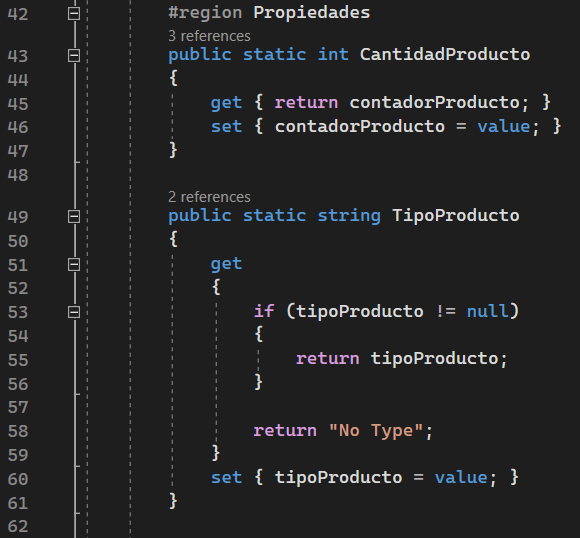
Al utilizar la sobrecarga en estos métodos me permite tener mas de una forma de llamar a estos metodos, en este caso mostrando mas o menos datos del objeto.



**+ Propiedades e Indexadores:**

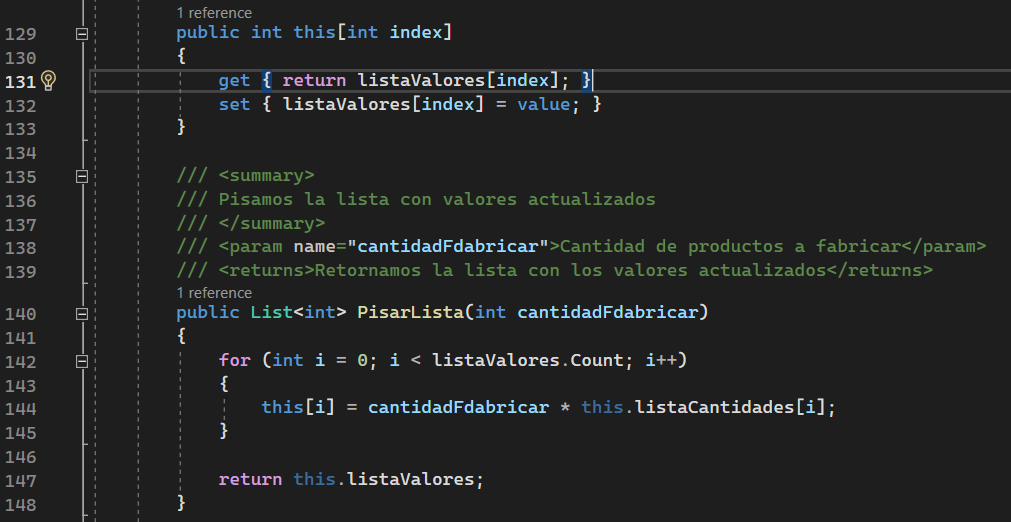
Utilizo propiedades en las clases Usuario, Operario y Supervisor - y luego en Empresa, Producto, VideoCard, Motherboard, Cabinet y Ram.





Utilizo propiedades para poder obtener o modificar atributos de las clases de forma mas práctica y segura, sin necesidad de tener mis atributos públicos.

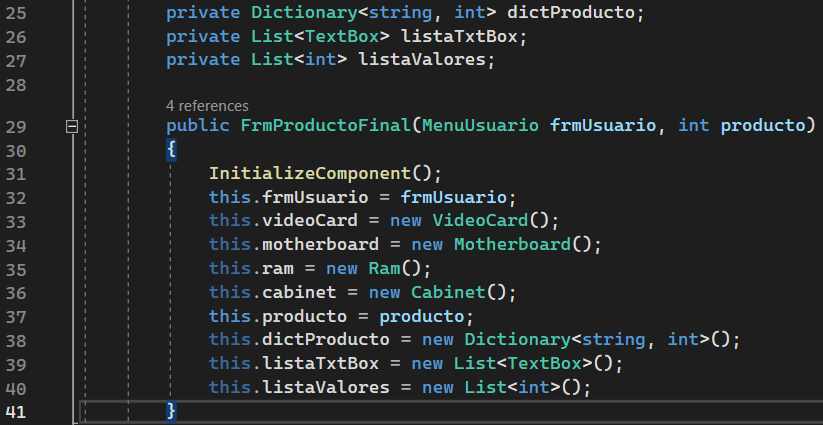
Utilizo indexadores en las clases VideoCard, Motherboard, Cabinet y Ram.



Utilizo indexadores para poder obtener o modificar índices específicos de colecciones, en mi caso, Listas.

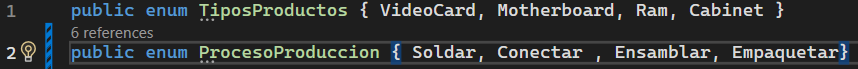
**+ Colecciones:**

Utilizo Listas y Diccionarios.



Lo que más utilizo en mi proyecto son listas, ya sea para datos hardcodeados, lista de valores (stock a actualizar), listas de objetos (text box, operarios, etc..) y los diccionarios lo utilizo más que nada para el Stock que voy modificando y mostrando en los formularios.

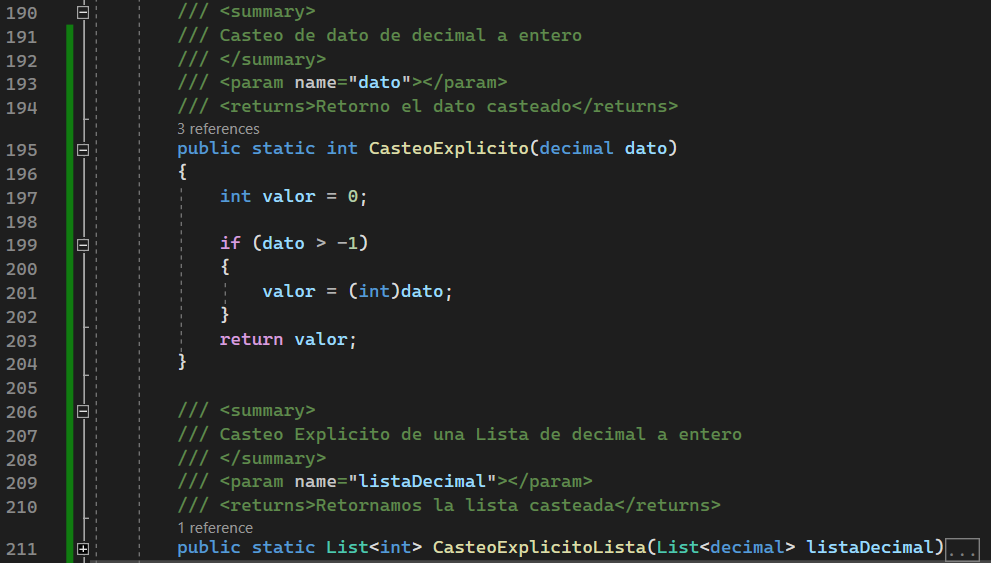
**+ Enumerados:**



Utilizo enumerados para trabajar con los tipos de productos y al momento de fabricar productos que estos tengan procesos para la producción.

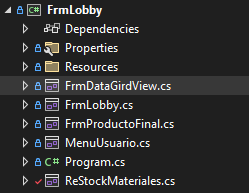
**+ Casteo Explicito:**

Convertimos de forma explícita un tipo decimal a int



**+ Formularios modales:**

En el proyecto utilizo cinco formularios



1.Lobby -> ingreso del usuario

2.Menu Usuario -> tiene la información y acceso a los sectores de producción.

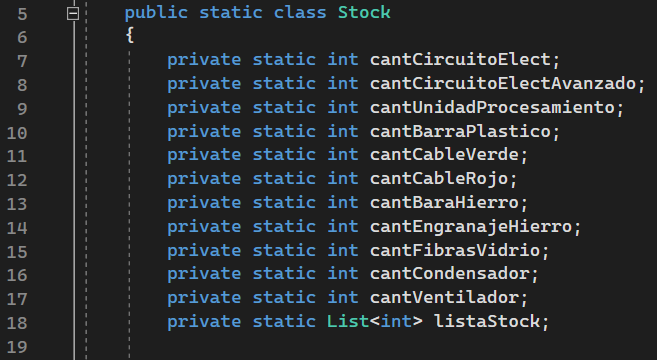
3.Sector Producción -> dependiendo con que botón interactúes en el menu anterior ingresaras al mismo form con sus diferencias de información para la producción de cierto producto.

4.Ingreso Stock -> siendo supervisor puede ingresar al formulario para modificar el sotck

5.Registro Operarios -> listado de los operarios ..

**+ Clases estáticas:**

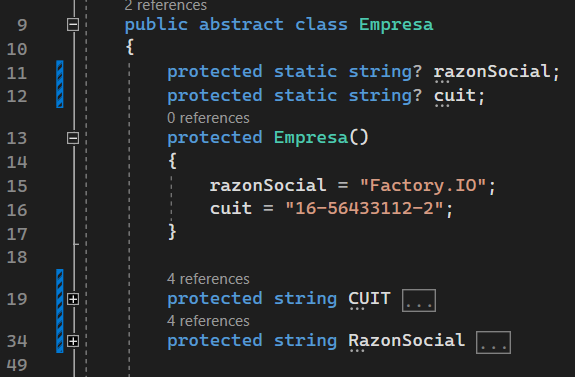
Hay dos clases estáticas en el proyecto Stock y Procesos



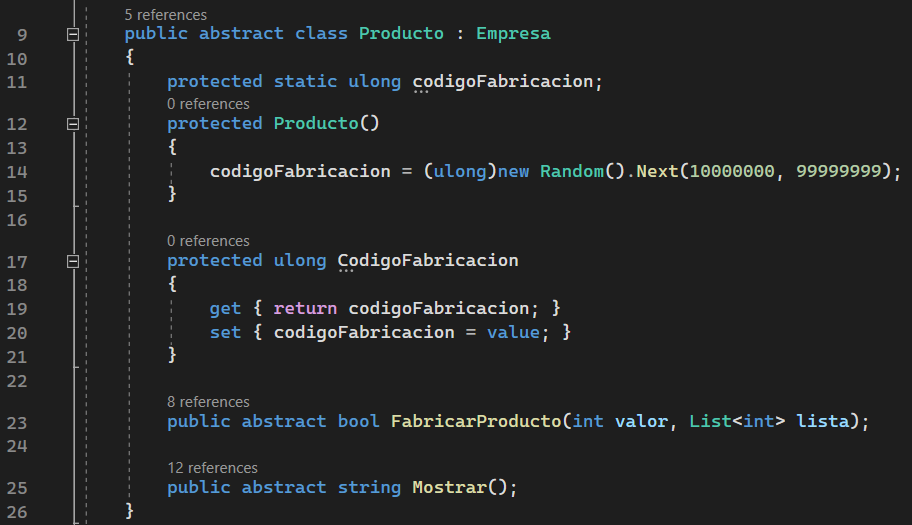
La que más "fuerza" tiene en el código es Stock que la utilizamos en gran parte del código ya que estamos interactuando con nuestro stock constantemente.

**+ Polimorfismo:**

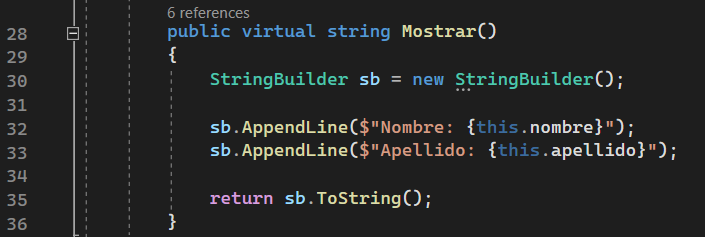
Clases abstractas y métodos abstractos



A la clase Empresa no hay necesidad de instanciarla en ningún momento y la clase Producto tampoco, pero si creamos dos métodos abstractos, uno para mostrar información y otro para fabricar producto.



Método virtual



En la clase Usuario le creamos un método virtual con el fin de sobrescribirlo en las clases Supervisor y Operario, teniendo así un método donde mostramos mas información que nuestra clase base.